DOCUMENT RESUME

ED 161 412 IR 006 249

AUTHOR Brahan, J. W.; Westrom, M. L.
TITLE NATAL-74; First Results.
PUB DATE Mar 78
NOTE 12p.; Paper presented at the Annual Meeting of the Association for the Development of Computer Based Instructional Systems (Dallas, Texas, March 1-4, 1978); For related documents, see IR 006 231 and 006 611

EDRS PRICE MF-$0.83 Plus Postage. HC Not Available from EDRS.
DESCRIPTORS *Computer Assisted Instruction; Computer Science Education; *Evaluation; *Instructional Design; Programing; *Programing Languages; Teacher Education
IDENTIFIERS *NATAL 74

ABSTRACT

NATAL is a high-level programming language designed to meet the requirements of the course author in the preparation of Computer Assisted Learning (CAL) materials. The basis for its design is found in the functional specification published by the National Research Council's Associate Committee on Instructional Technology; developers of the language also relied heavily on the 1969 EDUCOM report by Zinn, using his "aspects" for comparing programming languages as the basis for the definition of language requirements. The NATAL-74 project is outlined as follows: design goals, characteristics, implementations (course preparation system, file system, and run-time system), and preliminary evaluation. Education 491, a course on computers in secondary education, is taught to final-year education majors at the University of British Columbia. Six students from this course were permitted to do the NATAL-74 assignment, consisting of the construction of drill and practice programs in basic mathematics, rather than using a more limited CMI system. Results from this trial and other sources suggest that NATAL-74 meets the programming language requirements for CAL applications and can be an effective vehicle for the transfer of courseware between installations. (VT)

****
* Reproductions supplied by EDRS are the best that can be made *
* from the original document. *
****

ED161412

U S DEPARTMENT OF HEALTH,
EDUCATION & WELFARE
NATIONAL INSTITUTE OF
EDUCATION

THIS DOCUMENT HAS BEEN REPRODUCED EXACTLY AS RECEIVED FROM THE PERSON OR ORGANIZATION ORIGINATING IT. POINTS OF VIEW OR OPINIONS STATED DO NOT NECESSARILY REPRESENT OFFICIAL NATIONAL INSTITUTE OF EDUCATION POSITION OR POLICY

NATAL - 74; FIRST RESULTS

BY

J.W.Brahan

M.L.Westrom

"PERMISSION TO REPRODUCE THIS MATERIAL IN MICROFICHE ONLY HAS BEEN GRANTED BY

John J. Hirschbuhl

TO THE EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC) AND USERS OF THE ERIC SYSTEM."

IR006249

NATAL-74 : FIRST RESULTS

J.W. Brahan
National Research Council of Canada

M.L. Westrom
University of British Columbia

NATAL-74 is a high-level programming language designed to meet the requirements of the course author in the preparation of Computer-Aided Learning (CAL) materials. The basis of the NATAL-74 design is found in the functional specification(1) published by the National Research Council's Associate Committee on Instructional Technology. This document was prepared by a Working Group whose members came from centres across Canada where they were actively engaged in CAL projects. In producing the functio al specification, the working group members drew on their personal experience in a wide range of application areas with a variety of programming languages. In addition, the working group relied heavily on the 1969 EDUCOM report of Zinn(2), using his "aspects" for comparing programming languages as the basis for the definition of language requirements. Thus the functional specification was derived from input from a broad base of experienced users.

The detailed specification(3) for NATAL-74 was prepared by IBM Canada Ltd. under contract from NRC. Throughout this contract, the members of the Associate Committee on CAL Programming Languages provided comments and suggestions to the contractor. Late in 1974, the detailed specification was delivered and work started on the initial implementation of NATAL-74 on the DECsystem-10 computer which is used as the central computer facility in the National Research Council's CAL research project.

Design Goals

The fundamental objective of the NATAL-74 project is the development of a course-authoring language which meets user requirements and which permits the dissemination across system boundaries of computer-based course materials, without the cost of reprogramming those materials. Achievement of this objective is dependent on satisfying four major functional requirements

The first of these requirements is that the language must be based on a clearly defined standard which is capable of being implemented on the variety of computers which are used for CAL. This further implies the requirement that the language be defined in conformity with existing EDP standards issued by national and international standards bodies. To do otherwise would result in unnecessary and costly system programming changes in going from one manufacturers equipment to another.

A second main requirement is for the language to support the interactive applications common to CAL and which often involve the use of an extensive range of terminal equipment incorporating a variety of display-response capabilities. It should not be necessary for the user to change his program to permit it to run on a terminal which is different but functionally equivalent to the one on which it was developed. Any action required to adapt the user program in such a case should be carried out automatically by the CAL system.

A third requirement is that the language provide the computational, logical control and file-handling capabilities necessary to permit the variety of techniques which are encompassed by CAL. An effective course-authoring language is expected to support modes of operation which include adaptive tutorial instruction, drill and practice, operation as a problem-solving tool, simulation, information retrieval and operation in support of management of individualised instruction. Most applications incorporate more than one of these modes and to concentrate on any one at the expense of the others in the design of the language would limit its range of application.

The fourth main requirement is for the language to have a structure which will permit ease of learning and use by the CAL novitiate. Many course designers will not have an extensive computer science background and should not be expected to have to occupy themselves with details which can be performed for them by the system. Thus the language should incorporate the features and a structure which permits the course author to concentrate on the functional aspects of couseware design.

Characteristics of NATAL-74

NATAL-74 is a block-structured language which incorporates three fundamental elements: UNITS, PROCEDURES and FUNCTIONS. The UNIT is the element through which the user program communicates with the student. It may be considered as carrying out one complete instructional transaction. The UNIT provides for the display of

information, acceptance of input from specified devices, editing of input data, categorization of responses and implicit branching. The Display Sublanguage which is available in the UNIT provides flexibility in the formatting of alphanumeric information and the construction of complex graphic presentations. The UNIT has access to system general functions, edit functions and graphic functions and permits a highly complex exchange of information to be specified.

Program control, or instructional strategy, is embodied in the PROCEDURE which provides data gathering, data processing and logical branching capabilities. The PROCEDURE assembles information required by the UNIT, processes information generated by student responses, controls the sequence of instruction and maintains a record of program operation. A PROCEDURE may call other PROCEDURES or UNITS, allowing a powerful and flexible hierarchy to be constructed out of independent modules. The structure of a NATAL-74 program is illustrated in figure 1.

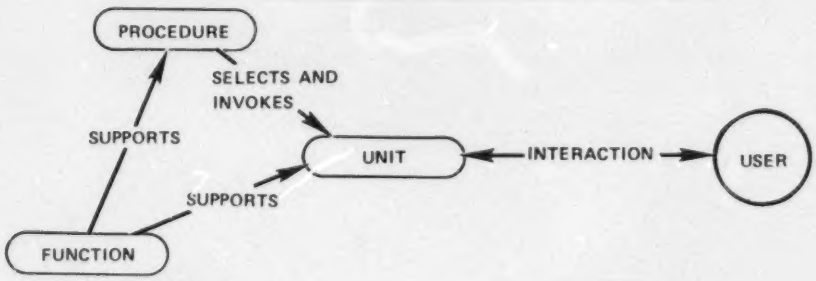

Figure 1
NATAL-74 STRUCTURE

NATAL-74 Implementations

The initial implementation of NATAL-74 is now running on the DECsystem-10 at the National Research Council. This implementation was undertaken with two main requirements in mind. The first was to get a NATAL-74 system in operation as soon as possible in order to start testing the language and get feedback from the organizations participating in the NRC CAL project. In response to this requirement, the decision was taken to avoid duplicating features in the NATAL-74 system which were provided by the DECsystem-10 monitor. Thus the implementation relies on the monitor for file-handling, user scheduling, and inter-process communication. The second requirement was to undertake the implementation in such a way that it could be transferred to

other machines with minimal effort. It was in response to this second requirement that BCPL was chosen as the implementation language and an attempt was made to avoid a high degree of dependence on features special to the DECsystem-10. There is obviously a certain amount of conflict between the two requirements and where system dependent features were introduced into the implementation they were to be identified to accomodate the migration of the system to other machines.

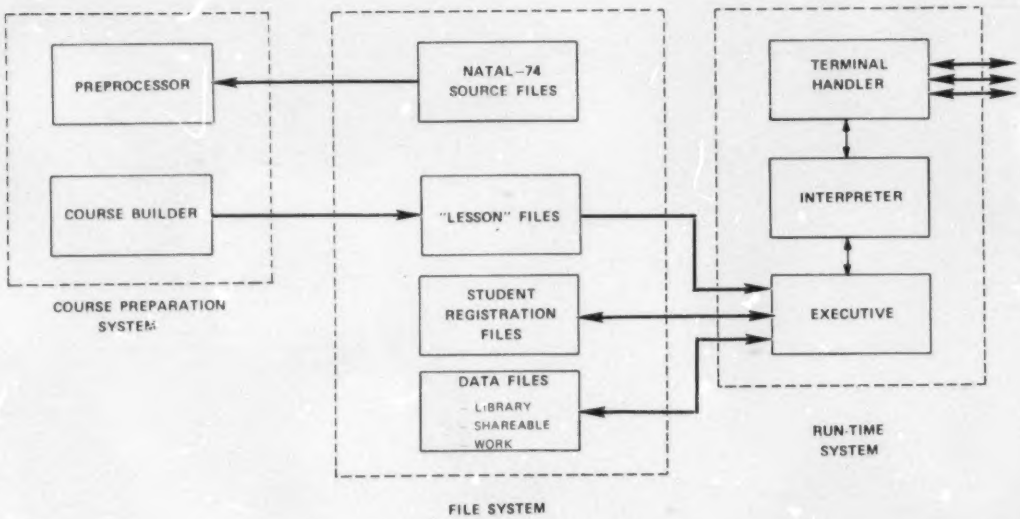

Figure 2
IMPLEMENTATION COMPONENTS

The implementation consists of three subsystems as illustrated in figure 2. The NATAL-74 source code is compiled into an intermediate language code by a preprocessor which also indicates errors in the user program which can be detected prior to execution. The preprocessed modules are assembled into courses by a builder program which carries out the functions of a linking loader. The run-time subsystem is made up of the interpreter, which executes the intermediate language code, the terminal handler, which supports the display sublanguage and links the subsystem to the user terminals, and the Executive, which links the subsystem to the host computer and controls access to shareable resources such as course variables and files. During the course of the implementation, no changes were to be made to the specification unless it could be clearly shown that there was an inconsistency which had to

be corrected, that the change would produce a significant saving in implementation cost or complexity with no functional change, that the specification deviated from widely accepted practice or that the specification did not meet a demonstrated user requirement. No changes were introduced simply to "improve the language". In other words, a concious effort was made to avoid redesigning the language during the implementation phase. Within these general groundrules, some changes were found to be necessary, but in terms of the overall specification, the total change has been relatively small. The areas of the specification most affected by changes are those dealing with graphics capabilities and file handling.

The implementation at present includes little in the way of what might be designated as author support facilities. In way of an example, to pass from a NATAL-74 source program to an operating course, the author must first create his source code program using one of the editors provided within the DECsystem-10 monitor, preprocess the source code modules using the preprocessor, then load them into a course using the builder before he can run the program. Much in the way of support facilities remains to be added to the system to relieve the user of the tedious detail involved in assembling, debugging and running a course. However, while the present support facilities are less than ideal, the NATAL-74 system is operational on the DECsystem-10 and is sufficiently complete that it can be used for effective testing of the language.

In April 1977, work started at the University of British Columbia (UBC) on the adaptation of the NATAL-74 implementation to the IBM 370/168 which runs under the Michigan Terminal System (MTS). The initial part of this undertaking concentrated on the run-time system. The major problems encountered in the transfer took the form of differences between the DECsystem-10 and the UBC versions of BCPL, and machine-dependent parameters in the interpreter and terminal handler which were not sufficiently well isolated and/or documented. As a result of the UBC project the machine-dependent elements of the implementation are now clearly identified and transfers to other machines will require considerably less effort.

Preliminary Evaluation

With a view to assessing the performance of NATAL-74 and its implementation in a variety of modes of operation, several CAL programs have been recoded from their original versions. These include CARE-1, written in Coursewriter-II at the Pennsylvania State University. This program provides a practical example of a tutorial mode application in which

there is extensive use made of multi-mode communication devices. MACMAN, a simulation of the heart-blood circulatory system developed at McMaster University has been recoded in NATAL-74 from its original FORTRAN-IV version. LUNA, a dialogue making extensive use of graphics, from the Physics Computer Development Project at the University of California has been partially recoded in NATAL-74. The work on this latter program has been particularly valuable since it brought to light a number of weaknesses in the graphics facilities in the original NATAL-74 specification. These have since been corrected and it is now possible for the course author to work in his own coordinate system, using the computer terminal to display all or part of the graphic work space he is using.

While existing CAL programs provide a good source of material for testing various aspects of NATAL-74, they have the disadvantage that often the design of the original program reflects limitations imposed by the original programming language and full advantage is not taken of the facilities offered in the higher level language. Thus the evaluation of NATAL-74 also requires use of the language in the development of original application programs. Within the NRC laboratory, several programs have been developed which illustrate the use of NATAL-74 in a number of application areas. A program has been written, for example, to illustrate the use of the computer in accessing the various documents which are required in commercial trading with countries throughout the world. This program drew heavily on the block structure, file handling and the graphics capabilities of NATAL-74. Another example is provided by a program designed to support the preparation and delivery of instructional and drill sequences used in a learner-controlled approach to second-language learning. Again, the block structure and file handling capabilities of NATAL-74 played an important role in the program design.

The first program development in NATAL-74 outside of the NRC laboratory took place at the University of British Columbia (UBC). During the first quarter of 1977, a single video display terminal at UBC was linked to the NRC computer to permit "field trials" with students in the Faculty of Education. The academic focus for this work on NATAL-74 was provided by a project concerned with the diagnosis of computation and algorithm errors made by children when learning to add, subtract, multiply and divide.

Education-491

Education-491, "Computers in Secondary Education," is an elective course at UBC open to education students in their final year. 491 students usually enrol for the

purpose of preparing themselves to teach Computing Studies in secondary schools. The course covers all aspects of the use of computers in secondary education, with emphasis on the teaching of Computing Studies and interactive applications (including CAL) in the schools. Students are required to have at least one full course in Computer Science as a prerequisite, but most have more background than this. It is known to be a "heavy" course so that it is reasonable to assume that most students are motivated and interested in computer applications in education.

The entire 491 class was taught the NATAL-74 language in four fifty-minute classes. Students were given a choice between doing an assignment using NATAL-74 or using a more limited CMI system (TAIM). The NATAL-74 assignment was considered to require more computing expertise because students would have to learn how to interact with the DECsystem-10 Monitor and the SOS Editor as well as learning how to construct, compile, and execute routines in NATAL-74. Furthermore, NATAL-74 would be available for only three months, and during the second month all students would be away doing student teaching. Six students were permitted to do the NATAL-74 assignment. All had a B.Sc. (or equivalent) with a major or concentration in Computer Science.

The Task

The task chosen was to construct Drill and Practice programs in addition, subtraction, multiplication and division of whole numbers and decimals. The drill exercises were generated so that they proceeded generally from easy to hard, and presented the variety of combinations of number length, carrying, and presence of zeros, that can occur in these operations; all under program control. The intention was to be able to generate basic operation problems of any particular type so that these could be provided for drill on that precise aspect of the algorithm and so that if a student consistently got such questions wrong, the portion of the algorithm he was misapplying could be specifically identified. The question generation routines were constucted as independent modules so that they could later be incorporated into more general drill and/or diagnostic programs. The diagnostic programs could then have appropriate remediation material added so that a comprehensive self-contained basic operation testing and teaching package for whole numbers and decimals could be built-up.

Developing The Courseware

The instruction on NATAL-74 was given during the month of January. Following the lectures, the students who were engaged in the NATAL-74 assignment were assembled to discuss the general approach to the problem. It was decided to divide the work into six distinct projects so that a minimum of interaction would be required between the students. The six projects were:

1: Addition of Whole Numbers.
2: Subtraction of Whole Numbers.
3: Multiplication of Whole Numbers.
4: Division of Whole Numbers.
5: Addition and Division of Decimals.
6: Subtraction and Multiplication of Decimals.

It was felt that the drill programs could be constructed as independent routines which could then be controlled by a higher-level PROCEDURE written later. Some time was spent initially designing the parameter lists for each of the routines and designing the question/response format for each of the different question types. The students, knowing the way in which their routines would be called, the standard format of the drill questions, and their specific subject area, could then proceed with the development of their routines independently.

Each student constructed a single UNIT for each of the question formats they would need. Some constructed General Functions to generate the types of operands they required - others generated their operands in their controlling PROCEDURE. Each student created a single controlling PROCEDURE for each operation.

As mentioned earlier, the NATAL-74 System was available for only three months. The first month passed while students learned the NATAL-74 language, explored the task and designed the question formats and calling protocols. During the second month students were engaged in student teaching. This left only the month of March to learn to use the DECsystem-10 Monitor, the SOS Editor, the NATAL-74 system procedures, and to design, code and test their NATAL-74 routines. Of course students had other duties at this time with 491 being only one of the courses they were taking.

During the coding of their NATAL-74 routines, students encountered a number of problems. Most of these had to do with learning to use the Monitor and Editor. Some arose because of discrepancies between the Author Guide definition of NATAL-74 and the actual implementation. At that time the NATAL-74 implementation was incomplete. On several occasions, because of missing functions, students had to

re-design their algorithms and then re-code them.

Even with these handicaps, all students were able to complete their assignments. Further, the Drill and Practice exercises they prepared were of high quality. There were no significant problems caused by "modularizing" the student assignments. Students were able to proceed independently in designing their algorithms for question generation and control. Standardizing the calling protocol (parameter lists) for each question type and the general question format was the only common consensus required. Lower level design did not demand further communication between the course authors.

Results

A specification has been developed and used as the basis of a running implementation of NATAL-74. While work on the implementation and initial testing has brought to light some weaknesses in the original "paper" specification, this was to be expected and modifications have been made which correct the deficiencies. While the specification has not yet reached the stage of a firm standard in that additional changes are expected to result from further testing, it should stabilize over the next few months.

The NATAL-74 projects which have been undertaken to date have demonstrated the possibilities of the language in CAL applications which include tutorial, drill-and-practice, simulation, and prompted authoring techniques. A good level of adaptability to various terminal types has been achieved. The DECsystem-10 implementation supports hardcopy terminals such as the Teletype and Multiwriter, alphanumeric display terminals such as the Lektromedia-103 and the Vucom-I and III, multi-media terminals such as the Lektromedia-120 which incorporates a defineable character set, slide projector, audio output and graphic input in addition to the conventional alphanumeric display capability, graphic terminals such as the Tektronix 4012 and 4051 and an experimental graphics terminal developed in the NRC laboratories. Using the automatic formatting capabilities of NATAL-74 for alphanumeric displays and the graphics function capabilities of the display sublanguage it has been shown possible to run programs on functionally equivalent terminals with no changes required on the part of the user.

The use which was made of NATAL-74 in the Education-491 class was limited to the preparation of drill and practice routines, but clearly demonstrated that the language could be quickly learned and applied to produce practical CAL materials. The capability of "modularizing" NATAL-74 code has a number of advantages. After preliminary

interface agreement, authors can proceed independently to define their routines. In the 491 trials, the student's primary job was to constuct one or two UNITS and a controlling PROCEDURE. They did not have to concern themselves with how their routines would eventually fit into the whole course. Yet the modularization permits an easy assembly of the various routines into a cohesive whole. Education-491 students found the syntax of NATAL easy to master and use.

Conclusions

While several features remain to be incorporaced in the NATAL-74 implementation and some additional specification changes are to be anticipated as a result of continued testing, the results achieved to date suggest that NATAL-74 meets the programming language requirements for CAL applications and can be an effective vehicle for the transfer of courseware between installations. The results support the continued investigation of NATAL-74 implementations on additional machines, including the development of mini-computer based delivery systems.

References

1. ACIT Working Panel. "A Functional Specification for a Programming Language for Computer-Aided Learning Applications", Report No. NRC-13659, Associate Committee on Instructional Technology, National Research Council, Ottawa 1972

2. Zinn, K.L. "Comparative Study of Languages for Programming Interactive Use of Computers in Instruction", EDUCOM Research Memorandum RM-1469, 1969

3. Westrom, M.L. "National Author Language NATAL-74 Specification Manual", Report No. NRC-14245, Associate Committee on Instructional Technology, National Research Council, Ottawa, 1974